ACADIA: Efficient and Robust Adversarial Attacks Against Deep Reinforcement Learning

Haider Ali[†], Mohannad Al Ameedi[†], Ananthram Swami[*], Rui Ning[**], Jiang Li[⋆], Hongyi Wu[‡], Jin-Hee Cho[†]
[†]Computer Science, Virginia Tech, VA, , {haiderali, mohada4, jicho}@vt.edu
[*]Army Research Laboratory, MD, USA, ananthram.swami.civ@army.mil
[**] Computer Science, Old Dominion University, VA, USA, rning@odu.edu
⋆ Electrical Engineering, Old Dominion University, VA, USA, jli@odu.edu
[‡]Electrical and Computer Engineering, The University of Arizona, AZ, USA, mhwu@arizona.edu

***Abstract*—Existing adversarial algorithms for Deep Reinforcement Learning (DRL) have largely focused on identifying an optimal time to attack a DRL agent. However, little work has been explored in injecting efficient adversarial perturbations in DRL environments. We propose a suite of novel DRL adversarial attacks, called ACADIA, representing AttaCks Against Deep reInforcement leArning. ACADIA provides a set of efficient and robust perturbation-based adversarial attacks to disturb the DRL agent's decision-making based on novel combinations of techniques utilizing *momentum*, *ADAM optimizer* (i.e., Root Mean Square Propagation, or *RMSProp*), and *initial randomization*. These kinds of DRL attacks with novel integration of such techniques have not been studied in the existing Deep Neural Networks (DNNs) and DRL research. We consider two well-known DRL algorithms, Deep-Q Learning Network (DQN) and Proximal Policy Optimization (PPO), under Atari games and MuJoCo where both targeted and non-targeted attacks are considered with or without the state-of-the-art defenses in DRL (i.e., RADIAL and ATLA). Our results demonstrate that the proposed ACADIA outperforms existing gradient-based counterparts under a wide range of experimental settings. ACADIA is nine times faster than the state-of-the-art Carlini & Wagner (CW) method with better performance under defenses of DRL.**

***Index Terms*—Deep reinforcement learning, adversarial attacks, Deep-Q learning Network, Proximal Policy Optimization**

I. Introduction

Deep Reinforcement Learning (DRL) algorithms learn policies to guide an agent to take optimal actions based on the states of its environment. They have successfully achieved high performance on various tasks, such as robotics, autonomous vehicles, and cybersecurity. Various attacks and defenses [7, 24] have been studied for supervised Deep Neural Network (DNN) applications, such as image classification [7] or natural language processing. However, adversarial attacks and defenses are largely unexplored in DRL settings. DRL is used in numerous critical safety and security applications and motivated us to develop robust DRL. To develop a robust DRL, it is natural to consider robust attacks for validating robust defenses in DRL.

Adversarial attacks in DRL have been studied to answer: *How to attack* and *When to attack*. The first *how-to-attack* question is about what perturbation method can be used to disrupt the state during an episode. The second *when-to-attack* question is about identifying an optimal time to attack during an episode. In this work, we focus on answering *how-to-attack*. To this end, we propose ACADIA, a set of novel adversarial AttaCks Against Deep reInforcement leArning. Therefore, we aim to develop robust and fast attacks by generating effective and efficient adversarial states in DRL settings.

Unlike DNN settings, developing fast and robust adversarial attacks in DRL faces the following challenges:

- *Sequential decision making under dynamic settings*: A dynamic series of steps exists for a DRL agent to continuously learn and take an action based on a reward. The DRL agent continuously tackles multiple situations in an episode. This implies that one time attack success in one step does not guarantee attack success in future steps.
- *Discrete or continuous action spaces*: Depending on a given setting, discrete or continuous action spaces are considered.
- *Lack of usefulness of DNN-based attacks in DRL settings*: The existing adversarial attacks developed for DNNs [1, 4, 5, 7, 19, 21, 22] do not guarantee their performance in DRL.

Our work has the following **key contributions**:

1) The proposed ACADIA provides a suite of efficient and effective adversarial attacks to disrupt DRL operations. The ACADIA consists of a suite of novel adversarial attacks: *Iterative ACADIA* (iACADIA), *ADAM-based Iterative ACADIA* (aiACADIA), and *Momentum-based, Iterative ACADIA* (miACADIA). These attacks are designed to generate fast and robust adversarial perturbations to compromise a DRL agent.
2) The iACADIA, aiACADIA and miACADIA are designed based on novel combinations of *RMSProp* (Root Mean Square Propagation) for iterative steps, *random initialization* for a random start, and *momentum* for escaping from local optima to boost efficiency and effectiveness in terms of the speed of performing attacks and the robustness of attacks under the state-of-the-art defenses in DRL.
3) We conduct extensive experiments based on Deep-Q Learning Network (DQN) and Proximal Policy Optimization (PPO) on Atari games (i.e., Pong, BankHeist, and RoadRunner) and MuJoCo environments with/without state-of-the-art defenses in a given DRL algorithm.
4) Our results show that all three variants of ACADIA outperform the Carlini & Wagner method (CW) on average attack execution time per perturbation (AET), while

maintaining comparable attack success rate (ASR), and average reward (AR), with miACADIA performing the best. ACADIA also outperforms the state-of-the-art perturbation methods on ASR and AR under DRL with defenses.

II. Related Work

A. Adversarial Attacks in DNNs

Goodfellow [7] proposed the Fast Gradient Sign Method (FGSM), which does not guarantee 100% ASR against state-of-the-art defenses. The CW method [1] is known to be the most effective method with 100% ASR but is significantly slower than other counterparts. Naïve FGSM provided the basis to build more sophisticated and better variants of FGSM in ASR, such as Randomized FGSM (RFGSM) [19], Diversity Iterative FGSM (DI-FGSM) [22], Momentum Iterative FGSM (MI-FGSM) [4], and ADAM Iterative FGM (AI-FGM) [21]. The Projected Gradient Descent (PGD) [5] was proposed as a variant of Iterative FGSM (I-FGSM) to enhance its robustness. AutoAttack [3] extended the PGD attack to resolve the suboptimal step size and problems of the objective function. AI-FGM was proposed for black-box attacks on DNNs [23].

Based on our extensive literature review, MI-FGSM and PGD attacks are considered to be the most efficient and robust state-of-the-art adversarial examples in DNNs. AI-FGM, RFGSM, DI-FGSM, AutoAttack and MI-FGSM were designed and evaluated in the context of DNNs. However, their performances have not been validated in DRL settings.

B. Adversarial Attacks in DRL

Huang, et al. [8] extended the adversarial attacks to DRL first by crafting the existing FGSM to algorithms like DQN to all time steps during an episode. Kos and Song [9] investigated the effectiveness of adversarial examples and random noise on DRL policies. Later, the state-of-the-art DRL attacks mainly identified an optimal time to attack the DRL agent [12]. Sun et al. [18] proposed a strategic timed attack based on [12]. They perturbed in a minimum number of critical moments to introduce severe damage to an agent while being undetectable. Tretschk et al. [20] used Adversarial Transformer Network (ATN) to maximize the adversarial reward through a sequence of adversarial inputs. Most of the *when-to-attack* strategies based attacks employed the CW [1] to perturb those critical moments. However, CW is known to be very slow. Chen et al. [2] proposed a common dominant adversarial examples generation method to confuse the agent with obstacles. Pan et al. [16] exploited the temporal consistency of the states. None of these attacks is generic enough to be widely used across varied settings of DRL since they show performance on limited settings. Fast perturbation methods, such as naïve FGSM [7], have been used in DRL [8]. However, naïve FGSM is easily detectable. Another variant of FGSM, called Projected Gradient Method (PGD), is one of the state-of-the-art DRL attacks and provides faster and higher ASR than CW under defenses. However, the state-of-the-art defenses in DRL [6, 15, 26] recently challenged the robustness of PGD-based attacks [5], which were originally designed for DNNs. Pattanaik, et al. [17] proposed Gradient-Based (GB) attacks similar to FGSM but with a different cost function. GB is also not robust due to lack of randomness. Therefore, there is a critical need to develop scalable, effective, and robust state perturbation attacks in DRL settings.

III. Preliminaries

A. Deep Reinforcement Learning

Reinforcement learning (RL) algorithms optimize the expected cumulative reward by training a policy π. This policy can be a deterministic or probabilistic function that maps state s to action a, $\pi : S \rightarrow A$, where S and A are state and action spaces, respectively. In Deep RL (DRL), this policy function π is learned by a neural network. Deep Q-Networks (DQN) [13] and Proximal Policy Optimization (PPO) are two well-known DRL algorithms, which are evaluated in our work.

B. Adversarial States Generation Methods

We leveraged the following methods to develop our attacks.

1) **Iterative Fast Gradient Sign Method (I-FGSM):** I-FGSM [10] takes multiple small steps of size α in the direction of the gradient. Given $x^0 = 0$, on every iteration k, it performs:

$$x^k = x^{k-1} - \text{clip}_\epsilon(\alpha \cdot \text{sign}(\nabla \text{loss}(x^{k-1}))). \tag{1}$$

2) **Randomized FGSM (RFGSM):** RFGSM [19] adds a small random step, α, to FGSM to escape the non-smooth vicinity of the data point before linearizing the model's loss. The perturbed state is computed as:

$$x^{\text{adv}}_{RFGSM} = x' + (\epsilon - \alpha) \cdot \text{sign}(\nabla_{x'} \text{loss}(x', y_{\text{true}})), \tag{2}$$

$$\text{where} \quad x' = x + \alpha \cdot \text{sign}(\mathcal{N}(0, I)). \tag{3}$$

3) **Momentum Iterative FGSM (MI-FGSM):** MI-FGSM [4] is a variant of FGSM that integrates the *momentum* at each step of an I-FGSM to escape from poor local maxima and stabilize update directions. Starting with $x^0 = 0$, at every iteration k, the perturbation update is:

$$g^k = \mu \cdot g^{k-1} + \frac{\nabla \text{loss}(x^{k-1})}{||\nabla \text{loss}(x^{k-1})||_1}, \tag{4}$$

$$\text{where} \quad x^k = x^{k-1} - \text{clip}_\epsilon(\alpha \cdot \text{sign}(g^k)). \tag{5}$$

Here μ is the decay factor and g^k is the accumulated gradient at iteration k.

C. Definitions

- **Non-Targeted Attack in DRL:** The attacker aims to reduce the reward of the DRL agent by crafting a perturbation that can lead to a sub-optimal action.
- **Targeted Attack in DRL:** The attacker seeks to perturb the state so that the DRL agent takes a specific targeted action. For example, in Atari Pong, a targeted attack could be to make the DRL agent take the action "up" at a given point.
- **Robustness of DRL Attack:** A DRL attack is said to be robust if it achieves high ASR and low AR under defenses.

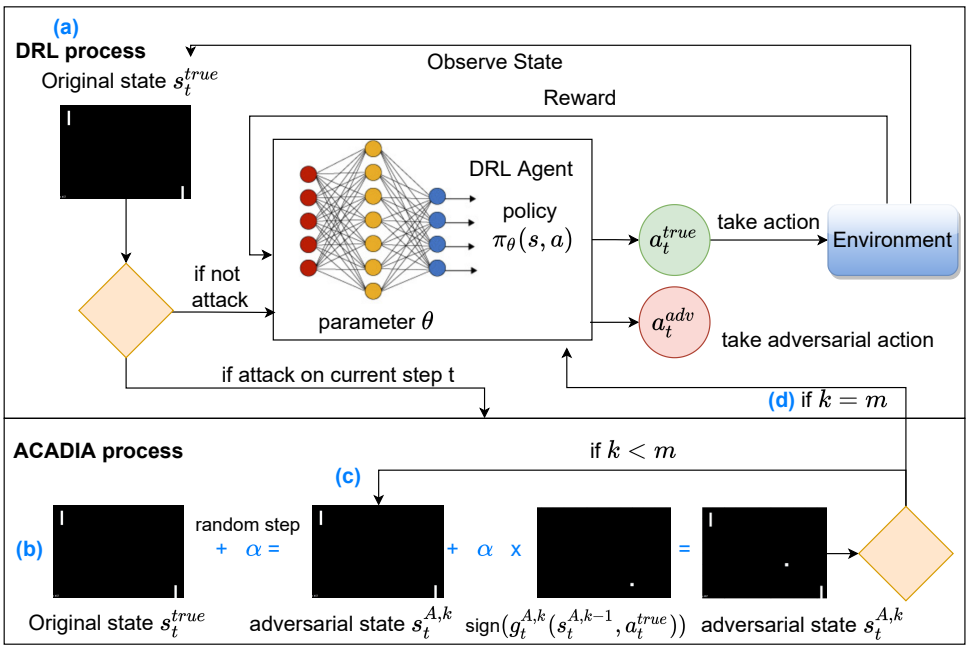

Fig. 1. An overview of Generic ACADIA. It can be either iACADIA, miACADIA or aiACADIA depending upon gradient g^A in (c). It attacks a frame t during an episode to compromise a DRL agent following: (a) DRL agent observes a true non-adversarial state s_t^{true} and takes non-adversarial, true action, a_t^{true}; (b) Start by adding a random step of size α to s_t^{true}; (c) Starting with $k = 0$ until number of steps m, compute the adversarial state $s_t^{\text{A},k}$ by calibrating either basic gradient $g_t^{\text{iACADIA},k}$, momentum-based accumulated gradient $g_t^{\text{miACADIA},k}$ or ADAM based gradient $g_t^{\text{aiACADIA},k}$, using $s_t^{A,k}$ and a_t^{true}, and then clipping it; and (d) Compute adversarial action, a_t^{adv}, by giving a final adversarial state $s_t^{\text{A},m-1}$ to the DRL agent.

IV. Problem Statement

A DRL agent interacts with the environment and learns policy π to choose action a given state s and obtains reward $r(s,a)$. This policy π can be a probabilistic model $\pi(\text{s},\text{a}) \sim [0,1]$, which gives the probability of taking action a given state s. The π can also be a deterministic model $s : a = \pi(\text{s})$. The goal of the DRL agent is to maximize the cumulative reward R_o by learning an optimal policy π^*. The reward that the DRL agent aims to maximize is the expected discounted reward over next $T-1$ time steps, represented by:

$$R_o = \sum_{t=0}^{T-1} E_{a_t \sim \pi(s_t)} \Big[\gamma^t r(s_t, a_t) \Big], \tag{6}$$

where $\gamma \in [0,1)$ is the discount factor. An attacker aims to reduce reward R_o by adding perturbation δ_t to the agent's observation s_t to mislead the agent to take non-optimal action a_t. The attacker has to generate perturbation δ as small as possible to be undetected. At each time step, the attacker may or may not choose to add perturbation δ to state s_t based on its strategy. In this way, the expected cumulative reward after the attack is given by:

$$R_{\text{adv}} = \sum_{t=0}^{T-1} E_{a_t^{\text{adv}} \sim \pi(s_t + u_t\, \delta_t)} \Big[\gamma^t r(s_t, a_t) \Big], \tag{7}$$

where u_t at given time t is 1 if the attacker injects perturbation; otherwise, $u_t = 0$.

In this work, we aim to solve the following problems:

- We aim to design the perturbations $\delta_0, \delta_1, \ldots, \delta_{T-1}$ to force the DRL agent to take the corresponding adversarial actions, denoted by $a_0^{\text{adv}}, a_2^{\text{adv}}, \ldots, a_{T-1}^{\text{adv}}$.
- These adversarial actions should reduce the reward of the DRL agent R_o, and the reward of the DRL agent under defense, R_{defense} in a non-targeted setting. The reduced reward is represented by R_{adv} in Eq. (7).
- These adversarial perturbations should generate the required targeted actions in the targeted attack setting.
- We aim to find each perturbation δ_t in a realistic time that could work under real settings.

V. Description of ACADIA

To develop robust, effective, and efficient perturbations, we propose the ACADIA framework that provides the following three novel attacks: iACADIA for Iterative ACADIA, aiACADIA for ADAM-based iACADIA, and miACADIA for Momentum-based iACADIA. ACADIA integrates *RMSProp*, *momentum*, *random initialization* and FGSM to maximize the effectiveness and efficiency of the proposed adversarial attacks in DRL settings. The ACADIA performs state perturbation attacks on observation and reward of the DRL agent. All of our attacks differ in computing the gradient of the loss function and changing a step size through a number of steps. Now we discuss a generic version of our attacks where we will attack a single state during an episode. We describe the generic form of ACADIA and the three schemes in detail as follows:

- *Generic* ACADIA: Let A be one of the ACADIA schemes (i.e., iACADIA, miACADIA or aiACADIA) which differs

in computing a gradient g^A of the loss function. We consider the true label, a_t^{true}, the action produced by the DRL policy at a time step t during an episode, represented by:

$$a_t^{\text{true}} = \pi(s_t^{\text{true}}), \tag{8}$$

where s_t^{true} is the original non-adversarial state at time step t. At each iterative step k of the attack until step m, we calculate adversarial state $s_t^{\text{A,k}}$ by taking a step of size α in the direction of the gradient $g_t^{A,k}$. Starting with adding the random step of size, α, to s_t^{true}, called a *random initialization*, $s_t^{A,0}$, which is given by:

$$s_t^{A,0} = s_t^{\text{true}} + \alpha \cdot \text{sign}(\mathcal{N}(0, I)), \tag{9}$$

on every iteration k of ACADIA until m, the attacker computes its state at k, $s_t^{A,k}$ by:

$$s_t^{A,k} = s_t^{A,k-1} + \alpha \cdot \text{sign}(g_t^{A,k}). \tag{10}$$

The attacker will compute its action at a frame t, a_t^{adv}, by:

$$a_t^{\text{adv}} = \pi(s_t^{\text{A,m}-1}). \tag{11}$$

Fig. 1 describes the key procedures of the generic `ACADIA` framework.

- `iACADIA`: This attack computes a regular gradient without incorporating any *momentum* or *RMSProp* term by:

$$g_t^{\text{iACADIA,k}} = \nabla_{s_t^{A,k-1}} \text{loss}(s_t^{A,k-1}, a_t^{\text{true}}), \tag{12}$$

where A is `iACADIA` here and loss function can be the cross entropy or Mean Squared Error (MSE). We use the cross entropy loss in our experiments.

- `miACADIA`: This attack incorporates *momentum* in computing a gradient with $g_t^{\text{miACADIA},0} = 0$ by:

$$g_t^{\text{miACADIA},k} = \tag{13}$$
$$\mu \cdot g_t^{\text{miACADIA},k-1} + \frac{\nabla_{s_t^{A,k-1}} \text{loss}(s_t^{A,k-1}, a_t^{\text{true}})}{||\nabla_{s_t^{A,k-1}} \text{loss}(s_t^{A,k-1}, a_t^{\text{true}})||_1},$$

where A is `miACADIA` here, μ is a decay factor and $g_t^{\text{miACADIA},k}$ is an accumulated gradient incorporating *momentum* at iteration k.

- `aiACADIA`: This attack uses the ADAM optimizer that incorporates *momentum* m and *RMSProp* v in computing a gradient with $m_t^{A,0} = 0$ and $v_t^{A,0} = 0$ and is performed by:

$$g_t^{A,k} = \frac{\nabla_{s_t^{A,k-1}} \text{loss}(s_t^{A,k-1}, a_t^{\text{true}})}{||\nabla_{s_t^{A,k-1}} \text{loss}(s_t^{A,k-1}, a_t^{\text{true}})||_1}, \tag{14}$$

$$m_t^{A,k} = \mu_1 \cdot m_t^{A,k-1} + (1 - \mu_1) \cdot g_t^{A,k}, \tag{15}$$

$$v_t^{A,k} = \mu_2 \cdot v_t^{A,k-1} + (1 - \mu_2) \cdot (g_t^{A,k})^2, \tag{16}$$

$$g_t^{\text{aiACADIA},k} = \frac{m_t^{A,k}}{\sqrt{v_t^{A,k} + \beta}}, \tag{17}$$

where A is `aiACADIA` here, μ_1 and μ_2 are decay factors and β is a very small number to make the denominator non-zero. This attack does not use the sign function in Eq. (10). Therefore, this equation can be re-written by:

$$s_t^{\text{aiACADIA},k} = s_t^{\text{aiACADIA},k-1} + \alpha \cdot g_t^{\text{aiACADIA},k}. \tag{18}$$

Removing the sign function here means adapting the step size α. This is exactly similar to adapting the learning rate to converge to global minima smoothly.

Eq. (10) means that we move policy π away from an optimal action by using the direction of the gradient. We make it iterative to take multiple gradient direction steps. Multiple gradient steps move the policy away from the optimal action, contributing to generating high ASR and low AR. Adding a random step α at the start contributes to high ASR under defense as it allows it to escape the non-smooth vicinity of the data point before linearizing the model's loss.

Eq. (13) shows the addition of a *momentum* term to the gradient at each step. This addresses the problem of the attack becoming stuck at a poor local maximum and gives the gradient a necessary boost, called *momentum*, to reach the global maximum. *Momentum* stabilizes the gradient updates. We observe in the experiments that *momentum* helps gain high ASR and low AR in the complex settings of DRL especially in targeted attacks and continuous environments of DRL. In summary, `miACADIA` uses the novel combination of *random start*, *iterative* steps of size α and *momentum* in a DRL setting.

The `miACADIA` may not converge with a constant step size of α. By incorporating *RMSProp*, we change the step size at every iteration for faster and smoother convergence. Hence, `aiACADIA` uses the novel combination of *random start*, iterative steps using *RMSProp*, and *momentum* in DRL.

Fig. 2 highlights the key differences between FGSM variants and our proposed `ACADIA`. *Random initialization* helps in robustness, and *momentum* together with *RMSProp* helps in smoother and faster convergence. FGSM-based perturbation methods are efficient and can work under realistic settings. Thus, we presented a novel perturbation method using these features to make it robust, effective and efficient. PGD (i.e., the baseline most similar to ours) differs from `ACADIA` in that `ACADIA` takes a fixed size step α in a random direction instead of uniformly choosing a random point. PGD does not incorporate *momentum* and *RMSProp*, showing poor convergence especially in targeted attacks and complex settings of DRL. PGD does have some *random initialization*, which increases robustness. AutoAttack, like other PGD-based attacks, has similar problems of convergence and is time-consuming as it is an ensemble of different PGD-based attacks and BlackBox attacks. The CW [1] is not only time-consuming but also not robust showing high detectability, as shown in our experiments. MI-FGSM does not incorporate *random initialization* and *RMSProp*, showing poor convergence and low robustness. `ACADIA` provides a comprehensive set of features to efficiently and effectively enhance performance under a wide range of DRL environments in attack, application, and defense types.

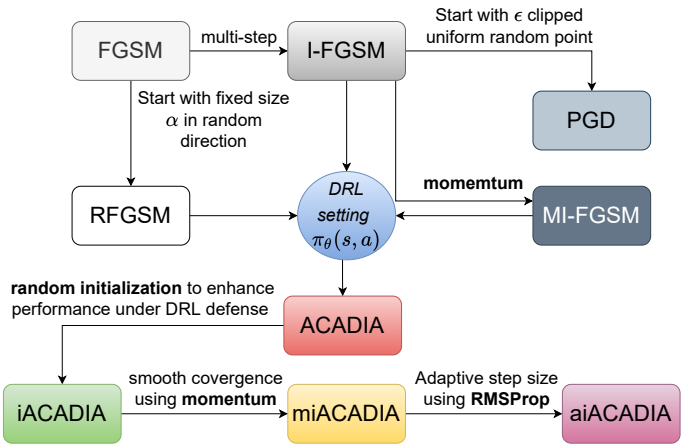

Fig. 2. Difference between FGSM variants and our proposed `aiACADIA`, `iACADIA` and `miACADIA`: Momentum and RMSProp improve performance, especially under complex settings of a targeted attack and continuous action space in DRL.

We do not employ any strategy to find an optimal time to attack during an episode as our major focus is to propose a comprehensive state perturbation attack in DRL (*how-to-attack*). Instead, we attack at every step during an episode. In addition, our approach allows a fair comparison with other state-of-the-art attacks because 'when-to-attack' can change the attack rate while we need to fix it to consider overall behavior across the whole episode. Thus, the number of attacks during an episode equals the episode length T (i.e., 100% attack rate).

VI. Experiment Setup

A. Setting for Attacks and Defenses

1) **Setting for Attacks:** We consider a white-box attack where the attacker does not have access to the training setup. However, at run time, the attacker has access to the trained DRL model and the state which it can perturb. In our experiments, we utilize trained DQN, and PPO models using the implementation of [14] and [25] to play within several environments. We use DQN for Atari games and PPO for MuJoCo Walker as the corresponding DRL algorithm is well-suitable for discrete and continuous environments, respectively.

We assume that an attacker has computational power equivalent to the commodity CPUs. To this end, we use the Google Colaboratory CPU (AMD EPYC 7B12, 2 CPUs @ 2.3 GHz, 13 GB RAM) for DQN experiments and personal computers for PPO due to the inability of MuJoCo to be run within the Google Colab. PPO experiments are run on Intel(R) Core(TM) i7-9750H CPU @ 2.60 GHz 32 GB RAM. We use the libraries of PyTorch and Open AI Gym.

We craft non-targeted and targeted attacks with and without defenses in our evaluation experiments. For targeted attacks, the state-of-the-art attacks of DRL use various strategies to find the optimal time throughout the episode. Instead of using such strategies, we generate targeted actions randomly to attack all time steps during an episode. This helps control the experiment across all attacks by having the strategy fixed. Attacking on all the time steps enables testing the attacks on all situations during the episode. We describe a generic evaluation experiment to apply attacks to DRL in Algorithm 1.

2) **Setting for Defenses:** Very few defenses have been proposed to defend against adversarial attacks in DRL. The conventional adversarial training defense has been widely used in DNNs [10] and DRL [9, 17]. However, recently more robust and efficient defense methods were proposed for DRL, such as Robust ADversarIAl Loss (RADIAL) [15], and State Adversarial (SA) [26], and Alternating Training of Learned Adversaries (ATLA) [25]. We consider RADIAL and ATLA which outperforms SA and other major defenses of DRL. In particular, we use RADIAL-DQN, RADIAL-PPO and ATLA-PPO as defenses. However, we could not implement ATLA-DQN due to the lack of resources given by [25].

B. Metrics

- *Average Attack Execution Time Per Perturbation (AET)* is the average time required to generate a perturbed state and is measured by $\text{AET}_T = \mathcal{T}(N_A)/N_S$, where N_S is the total number of adversarial states computed, $\mathcal{T}(N_A)$ is the total time elapsed to generate all adversarial states.
- *Average Reward (AR)* measures the average reward across all episodes. Given N_e, the number of episodes, and r_i, the accumulated reward during an episode i, AR is measured by $\text{AR} = \sum_{i=0}^{N_e} r_i/N_e$.
- *Attack Success Rate (ASR)* measures the total number of attack successes over the total number of attack attempts. ASR is measured by $\text{ASR}_{NT} = N^{AS}/N_{NT}$, where N^{AS} is the total number of attack successes by targeted or non-targeted attacks and N_{NT} is the total number of attempts by targeted or non-targeted attacks. Under non-targeted attacks, a failure indicates that the attack is entirely unable to succeed or the DRL agent takes an action maximizing its reward. An attack attempt is defined as a one-time attack per time step. On the other hand, targeted attack success means generating a perturbation targeting to lead the DRL agent to take the targeted action. Thus, reward reduction is not considered an attack success in targeted attacks.
- *ASR in continuous environments (ASR-C)* is our novel ASR metric in a continuous environment, such as MuJoCo. We estimate a Mean Absolute Error (MAE) of two actions. If the MAE of two actions is less than the threshold λ, we treat both actions as equal. In a non-targeted setting, an attempt is considered success if $MAE(a^{\text{true}}, a^{\text{adv}}) > \lambda$. However, in a targeted setting, success is defined when $MAE(a^{\text{targeted}}, a^{\text{adv}}) < \lambda$. We set $\lambda = 0.1$ in MuJoCo Walker experiments. We set $\lambda = 0.1$ in non-targeted setting because $\lambda \geq 0.1$ can create sufficient deviation from optimal actions.

C. Parameterization

The number of steps (m) is fixed at 20, which is optimal for FGSM-based perturbation attacks based on our empirical experiments. However, we use $m = 1,000$ for the CW method at which it performed the best. For a fair comparison, we also show results for CW with $m = 20$. We consider naïve FGSM, using $m = 1$ by definition, as a baseline. AET cannot

be fixed as it fluctuates with high variance. However, for a fair comparison, we consider CW with $m = 100$ and $m = 120$ as it will take AET comparable to ACADIA. By fixing $m = 20$, baseline perturbation methods are also taking AET comparable to ACADIA. We fix $\epsilon = 8/255$ and $\alpha = 2/255$ for our experiments under DQN and PPO, which are identified as optimal in terms of detectability, AR, and ASR after rigorous sensitivity analysis on all considered perturbation methods. However, our perturbation method can also work with even smaller ϵ's. Due to the lack of space, we have only shown the results of optimal parameters on different baselines for a fair comparison. We use the cross-entropy loss function of PyTorch for gradient-based attacks. We employ L_∞ norm in all FGSM-based baselines, including our attacks while using L_2 norm for the CW. We run every experiment 100 times and report the mean value and standard deviation of each metric.

D. Schemes for Comparison

We use the following perturbation attacks for comparison against our ACADIA variants. As discussed earlier, we do not consider strategic attacks based on *when-to-attack* [11, 18] as schemes for comparison because our work mainly concerns *how-to-attack*. We also did not consider AutoAttack [3] as our baseline because it does not perform well on targeted attacks. In addition, in DRL setting, AutoAttack was mostly similar to PGD, which is considered in this work. Our proposed ACADIA is compared against the following baseline and state-of-the-art counterpart schemes:

- **Fast Gradient Sign Method (FGSM)** [7]: We choose this as our baseline scheme because it has been extensively used in both DNN and DRL settings, such as *uniform attack* [8].
- **Carlini & Wagner Method (CW)** [1]: This method is one of the well-known state-of-the-art methods in DRL settings, guaranteeing 100% ASR. The state-of-the-art end goal-based DRL attacks [11, 18] use the CW to generate targeted perturbations. Hence, outperforming the CW is a clear indicator of advancing the state-of-the-art attacks in DL.
- **Projected Gradient Method (PGD)** [5]: PGD is considered to be the most robust and fastest state-of-the-art attack in DRL settings. Due to its robustness, PGD is usually employed by defenders to test their defenses as in [15, 26].
- **Momentum Iterative Fast Gradient Sign Method (MI-FGSM)** [4]: This is the state-of-the-art adversarial example generator in DNN settings.
- **Diversity Iterative Fast Gradient Sign Method (DI-FGSM)** [22]: We also extend DI-FGSM from DNNs to DRL and compare it with our method.

We did not consider Robust Sarsa (RS) and Maximal Action Difference (MAD) attacks [26] as the counterpart schemes to be compared against ACADIA because our attacks depend on critic so we mainly consider critic-dependent schemes. Moreover, RS, MAD and the optimal adversary of [25] do not work under the SA and ATLA defenses as shown in [26] and [25].

We report the average and standard deviation of ARs, AETs and ASRs. We conduct extensive experiments to evaluate the efficiency, effectiveness, and robustness of our attacks compared to the benchmark attacks. We consider both targeted and non-targeted attacks, and DRL with and without the state-of-the-art defenses (i.e., RADIAL and ATLA). For non-targeted (NT) attacks, we avoid the desired action identified by the DRL model. For targeted (T) attacks, we take a particular perturbation to mislead a DRL agent to take the attacker's desired action. The targeted actions are chosen randomly to evaluate the ability of perturbation methods to generate a targeted action. We do not include targeted attacks on MuJoCo Walker environments since most of the attacks, including CW, cannot precisely generate random targeted actions. Therefore, we leave studying efficient and precise targeted attacks in continuous DRL environments for our future work. Our source code is available on these Github repositories: DQN and PPO MuJoCo experiments.

Algorithm 1 Perform Attacks in DRL

Input:
$A \leftarrow$ an actions set, $s_0 \leftarrow$ an initial non-adversarial state
Parameters:
targeted $\leftarrow$ return 1 if attack is targeted; 0 otherwise
defense $\leftarrow$ return 1 if defense is applied; 0 otherwise
procedure PERTURBATIONMETHOD((A, s_0)) ▷ a perturbation method used
  for each episode **do**
    for each step t during an episode **do**
      if targeted is true **then**
        $a_t^{\text{adv}*}$ = RandomStrategy(A)
        s_t^{adv} = PerturbationMethod(s_t, a_t^{adv*})
      else
        s_t^{adv} = PerturbationMethod(s_t)
      end if
      if defense is true **then**
        $a_t^{\text{adv}} = \text{DRL}_{\text{defense}}(s_t^{\text{adv}})$
      else
        $a_t^{\text{adv}} = \text{DRL}(s_t^{\text{adv}})$
      end if
      $r_t^{\text{adv}}, s_{t+1}$, done = Perform($a_t^{\text{adv}}$)
      if done is true **then**
        break
      end if
    end for
  end for
end procedure

VII. Results & Analyses

A. Attack Success Rate (ASR)

Table I shows a comprehensive comparison between ACADIA variants and the baselines on Attack Sucess Rate (ASR) when tested on DQN, RADIAL-DQN, PPO and RADIAL-PPO playing Atari and MuJoCo environments. Higher ASR means a stronger attack. It is quite obvious that our proposed ACADIA outperforms its counterparts and baseline schemes in ASR particularly under harsher environments with targeted attacks or defenses in DRL (i.e., RADIAL).

Table II also shows the comparison of all the schemes considered in ASR-Continuous (ASR-C) for MuJoCo Walker where a DRL agent uses ATLA-PPO and RADIAL-PPO defenses

TABLE I
ATTACK SUCCESS RATE (ASR) FOR DQN AND PPO UNDER TARGETED (T) AND NON-TARGETED (NT) ATTACKS WITH AND WITHOUT RADIAL DEFENSE ON ATARI GAMES AND MUJOCO ENVIRONMENTS. HIGHER ASR MEANS A STRONGER ATTACK.

Perturbation Method (steps)	Pong				BankHeist				RoadRunner				MuJoCo Walker	
	Vanilla-DQN		RADIAL-DQN		Vanilla-DQN		RADIAL-DQN		Vanilla-DQN		RADIAL-DQN		Vanilla-PPO	RADIAL-PPO
	NT	T	NT	T	NT	T	NT	T	NT	T	NT	T	NT	NT
CW (1000)	100%	100%	3%	16%	100%	100%	3%	5%	95%	97%	1%	0%	100%	98%
CW (120)	100%	16%	3%	17%	94%	4%	0%	6%	83%	7%	3%	4%	98%	100%
CW (100)	100%	17%	3%	17%	91%	5%	0%	4%	94%	7%	2%	5%	96%	100%
CW (20)	100%	16%	3%	16%	100%	6%	3%	5%	98%	6%	2%	0%	100%	98%
PGD (20)	100%	100%	99%	72%	100%	100%	99%	78%	99%	76%	99%	89%	100%	100%
DIFGSM (20)	100%	99%	73%	44%	100%	99%	88%	68%	99%	79%	89%	79%	100%	100%
MIFGSM (20)	100%	99%	75%	65%	100%	100%	100%	88%	99%	96%	100%	91%	100%	100%
FGSM (1)	85%	35%	28%	16%	100%	66%	47%	47%	79%	43%	91%	85%	100%	100%
miACADIA (20)	**100%**	**100%**	**99%**	**78%**	**100%**	**100%**	**100%**	**90%**	**99%**	**98%**	**100%**	**95%**	**100%**	**100%**
iACADIA (20)	**100%**	**100%**	**99%**	**73%**	**100%**	**100%**	**99%**	**79%**	**99%**	**76%**	**100%**	**91%**	**100%**	**100%**
aiACADIA (20)	**100%**	**100%**	**99%**	**75%**	**100%**	**100%**	**100%**	**92%**	**99%**	**97%**	**99%**	**96%**	**100%**	**100%**

TABLE II
ASR-CONTINUOUS (ASR-C) FOR MUJOCO WALKER USING PPO

Perturbation Method (steps)	Vanilla-PPO	ATLA-PPO	RADIAL-PPO
CW (1000)	100%	100%	98%
CW (20)	100%	82%	98%
CW (120)	98%	81%	97%
CW (100)	96%	80%	97%
PGD (20)	100%	98%	100%
MI-FGSM (20)	100%	98%	100%
FGSM (1)	100%	96%	100%
miACADIA (20)	**100%**	**100%**	**100%**
iACADIA (20)	**100%**	**98%**	**100%**
aiACADIA (20)	**100%**	**100%**	**100%**

in the presence of non-targeted attacks. This metric actually shows whether an attack can generate a different action or not based on the λ value. Here we set $\lambda = 0.1$. Clearly, ACADIA performs either better than or is comparable to baselines. CW is affected under RADIAL-PPO but with a minor impact. Hence, interestingly, CW works well under defenses in the MuJoCo PPO experiments while performing poorly on Atari games under RADIAL-DQN.

B. Average Reward (AR)

Table III shows the AR performance comparison of all attack methods. Lower AR means a stronger attack. ACADIA variants outperform all baselines. When RADIAL is used as a defense in DRL, all alternatives are impacted. However, the ACADIA variants are impacted the least. On the other hand, CW completely fails in both non-targeted and targeted attacks, performing worse than even naïve FGSM under RADIAL defense. PGD performs comparably to the worst ACADIA variant in ASR but not to the best variant of ACADIA under a wide range of environmental settings (i.e., targeted or non-targeted attacks and with/without defense).

Table III shows the performance of all the considered schemes in AR under 14 different environments. aiACADIA outperforms CW (1000) under 10 out of 14 environments. CW (1000) only outperforms aiACADIA under 4 environments which do not consider RADIAL defense. miACADIA and aiACADIA outperform MI-FGSM in AR in 11 out of 14 environments. The remaining three cases show miACADIA and aiACADIA show comparable performance to MI-FGSM and CW(1000). CW(100), CW(120) and CW(20) perform worse than ACADIA in all cases in AR. Hence, overall our proposed attacks outperform in AR.

Table IV shows the performance comparison of all the compared methods in AR for MuJoCo Walker when DRL (i.e., PPO) uses ATLA or RADIAL as a defense under non-targeted attacks. ACADIA again show robustness against RADIAL-PPO because it achieves the minimum reward among all the baselines. ACADIA outperforms showing minimum rewards under ATLA-PPO as compared to the gradient-based counterparts. However, CW can achieve a lower reward than our attacks under ATLA-PPO. The performance of our attacks can be evaluated through the reward reduction obtained. In MuJoCo Walker, the maximum reward achieved by ATLA-PPO under no attack is around 4,000 while our best attack gives 321. Therefore, the reward reduction is around 3,679 by our attacks. Considering our attacks achieve 100% ASR, we can conclude that our attacks do not take the worst actions but can distract the DRL agent to take non-optimal actions, which fulfills our goal to take non-optimal actions with and without defenses.

C. Average Attack Execution Time (AET)

Table V shows the comparison of AET for Atari Pong played by DQN. We also performed experiments for Atari RoadRunner and Atari BankHeist played by DQN and MuJoCo Walker played by PPO under targeted and non-targeted attacks. Overall, FGSM variants perform comparably. Non-targeted iACADIA, the best performing, is 12 *ms* faster than the non-targeted PGD attack for Pong played by DQN. Thus, PGD can be considered equivalent to ACADIA in AET. For targeted attacks, PGD shows comparable performance to ACADIA. iACADIA outperforms again in AET for targeted attacks. Interestingly, CW with 20 steps is faster than most FGSM variants with 20 steps although CW performs poorly with 20 steps. CW with 100-120 steps takes AET comparable to ACADIA. miACADIA (20 steps) is six to nine times faster than the state-of-the-art CW (1,000 steps) with better robustness. Similar results are observed for other Atari games and MuJoCo environments. As our attacks take a little more time than other FGSM counterparts, the question is whether it is worth spending more time. Minute

TABLE III
AVERAGE REWARD (AR) FOR DQN AND PPO UNDER TARGETED (T) AND NON-TARGETED (NT) ATTACKS WITH AND WITHOUT RADIAL DEFENSE ON ATARI GAMES AND MUJOCO ENVIRONMENTS. LOWER AR MEANS A STRONGER ATTACK.

Perturbation Method (steps)	Pong				BankHeist			
	Vanilla DQN		RADIAL-DQN		Vanilla DQN		RADIAL-DQN	
	Non-Targeted	Targeted	Non-Targeted	Targeted	Non-Targeted	Targeted	Non-Targeted	Targeted
CW (1000)	−21 ± 0	−21 ± 0	+20.85 ± 0.3	+20.5 ± 0.5	0 ± 0	0 ± 0	252 ± 34	302 ± 39
CW (120)	−21 ± 0	+20.5 ± 0.5	+21 ± 0	+20.5 ± 0.5	630 ± 0	780 ± 0	390 ± 390	390 ± 170
CW (100)	−19 ± 0	+21 ± 0	+20.5 ± 0.5	+20.5 ± 0.5	450 ± 0	770 ± 10	390 ± 170	390 ± 170
CW (20)	−21 ± 0	+20.7 ± 0.4	+20.7 ± 0.4	+20.8 ± 0.4	550 ± 50	710 ± 10	309 ± 36	321 ± 46
PGD (20)	−21 ± 0	−20.6 ± 0.4	−20.9 ± 0.2	−20.4 ± 0.8	6 ± 9	10 ± 0	2.3 ± 5.5	4 ± 6
DIFGSM (20)	−21 ± 0	−20.3 ± 0.6	−19.8 ± 1.3	−16.7 ± 2.6	3 ± 4	20 ± 8	1 ± 4	4 ± 6
MIFGSM (20)	−21 ± 0	−20.8 ± 0.4	−20.5 ± 0.7	−20.4 ± 0.7	0 ± 0	20 ± 8	0.6 ± 2.4	3.6 ± 5.4
FGSM (1)	−21 ± 0	−20.7 ± 0.9	+20.7 ± 0.4	+16.8 ± 7.8	0 ± 0	23 ± 20	0 ± 0	2.3 ± 4.2
miACADIA (20)	**−21± 0**	−20.7± 0.4	**−20.9± 0.2**	−20.3± 0.9	**0± 0**	16 ± 17	**1.6± 4.5**	3± 5.2
iACADIA (20)	**−21± 0**	−20.3± 0.6	**−20.9± 0.3**	−20.2± 0.9	**6± 4**	**6± 4**	**1± 3**	3± 4.5
aiACADIA (20)	**−21± 0**	−20.5± 0.5	**−21± 0**	−21± 0	**0± 0**	20 ± 8	**1± 2**	3± 0

Perturbation Method (steps)	RoadRunner				MuJoCo Walker	
	Vanilla DQN		RADIAL-DQN		Vanilla PPO	RADIAL-PPO
	Non-Targeted	Targeted	Non-Targeted	Targeted	Non-Targeted	Non-Targeted
CW (1000)	0 ± 0	0 ± 0	14000 ± 1000	18400 ± 15400	−7 ± 3	−14 ± 1
CW (120)	11700 ± 0	0 ± 0	5750 ± 4750	9050 ± 5550	−7 ± 2	−11 ± 0.3
CW (100)	100 ± 0	0 ± 0	22200 ± 15000	9050 ± 5550	−6 ± 2	−12 ± 0.3
CW (20)	8600 ± 0	0 ± 0	14000 ± 1000	18400 ± 15400	−5 ± 8	−18 ± 3
PGD (20)	200 ± 141	0 ± 0	17 ± 45	23 ± 76	71 ± 3	−49 ± 1
DIFGSM (20)	1100 ± 816	0 ± 0	87 ± 106	280 ± 399	68 ± 8	−49 ± 1
MIFGSM (20)	500 ± 0	0 ± 0	3 ± 18	3 ± 18	76 ± 4	−49 ± 1
FGSM (1)	100 ± 0	0 ± 0	247 ± 394	220 ± 338	52 ± 3	−49 ± 1
miACADIA (20)	**33± 47**	**0± 0**	**0± 0**	73± 254	**52 ± 3**	**−50 ± 2**
iACADIA (20)	**100± 141**	**0± 0**	**13± 34**	33± 79	**67 ± 3**	**−50 ± 1**
aiACADIA (20)	**23± 22**	0 ± 0	**0± 0**	0± 0	**52 ± 3**	**−51 ± 3**

TABLE IV
AVERAGE REWARD (AR) FOR MUJOCO WALKER USING PPO UNDER NON-TARGETED ATTACKS

Perturbation Method (steps)	Vanilla-PPO	ATLA-PPO	RADIAL-PPO
CW (1000)	−7 ± 3	−14 ± 1	−14 ± 1
CW (20)	−5 ± 8	−7 ± 4	−18 ± 3
CW (120)	−7 ± 2	−10 ± 2	−11 ± 0.3
CW (100)	−6 ± 2	−10 ± 1	−12 ± 0.3
PGD (20)	71 ± 3	330 ± 42	−49 ± 1
MI-FGSM (20)	76 ± 4	331 ± 48	−49 ± 1
FGSM (1)	52 ± 3	351 ± 27	−49 ± 1
miACADIA (20)	52 ± 3	322 ± 80	−50 ± 2
iACADIA (20)	67 ± 3	325 ± 43	−50 ± 1
aiACADIA (20)	52 ± 3	321 ± 43	−51 ± 3

TABLE V
ATTACK EXECUTION TIME (AET) FOR PONG USING DQN

Perturbation Method (steps)	Pong	
	Non-Targeted	Targeted
CW (1000)	716 ± 32	963 ± 20
CW (120)	118 ± 9	119 ± 1
CW (100)	97 ± 5	99 ± 6
MIFGSM (20)	128 ± 8	138 ± 8
DIFGSM (20)	103 ± 6	135 ± 14
PGD (20)	94 ± 6	117 ± 13
CW (20)	21 ± 2	26 ± 2
FGSM (1)	6 ± 2	6.3 ± 0.7
miACADIA (20)	**126 ± 7**	**125 ± 7**
iACADIA (20)	**82 ± 6**	**98 ± 6**
aiACADIA (20)	**138 ± 6**	**141 ± 7**

increase of 20 *ms* and 2-10 *ms* in AET boosted ASR of miACADIA and aiACADIA significantly up to 22% better than PGD and up to 24% better than MI-FGSM. Overall our attacks are effective under realistic situations as the time to craft a perturbation in our attacks is only a few milliseconds.

We observe that CW can be 7 to 22 times slower than our attacks. Although in the MuJoCo experiments, CW takes less time than in the Atari environments, it still cannot work under realistic situations where we have less time to craft the perturbation. Overall we show that our attacks are effective under realistic situations as the time to craft a perturbation in our attacks is only a few milliseconds.

In AET, iACADIA is better than miACADIA and miACADIA is better than aiACADIA because miACADIA incorporates *momentum* and aiACADIA incorporates both *momentum* and *RMSProp*. Even our basic version, iACADIA, performs better than most of the baselines.

On ASR and AR metrics, we observed that aiACADIA and miACADIA perform similarly in most cases. aiACADIA outperforms in a few instances, such as when RoadRunner is played by Vanilla DQN. Therefore, incorporating *RMSProp* helps in these cases. However, using *momentum* in the attack significantly helps achieve high ASR and low AR. As miACADIA takes less time than aiACADIA and shows comparable performance on AR and ASR, miACADIA can be considered the best variant.

VIII. CONCLUSIONS & FUTURE WORK

We proposed ACADIA, a set of novel adversarial AttaCks Against Deep reInforcement leArning. The ACADIA includes *Iterative ACADIA* (iACADIA), *ADAM-based Iterative ACADIA* (aiACADIA), and *Momentum-based, Iterative ACADIA* (miACADIA). To develop these, we took a novel combination of *RMSProp* (Root Mean Squared Propagation) for iterative steps, *random initialization* for a random start, and *momentum*

for escaping from local optima to significantly enhance efficiency and effectiveness in terms of the speed of performing attacks and the robustness of attacks under defenses in DRL. Our work is the first to develop efficient and robust perturbations specifically for DRL. Our **key findings** are:

- ACADIA is deployable to time-sensitive real-life applications as it can generate the state perturbations in less than 140 ms, which is a realistic time.
- ACADIA is nine times faster than CW and comparable to their FGSM counterparts, which are known as the most efficient and effective state-of-the-art attacks in DL.
- ACADIA outperforms all considered attack methods in Attack Success Rate and Average Reward overall baselines, especially under the defense.
- Under no defense, ACADIA is either better or comparable to baselines. PGD could not perform well on targeted attack settings while CW could not perform well under defenses.
- miACADIA can be considered as our best variant in terms of efficiency and effectiveness.

Through our extensive experiments, we found some limitations in our ACADIA, which will be explored in our future research. First, our work has not addressed the attack based on *when-to-attack* strategies. Using these strategies did not allow us to compare our scheme with the existing *when-to-attack* counterparts. However, we strongly recommend using ACADIA with *when-to-attack* strategies as ACADIA can be detected if we will attack all time steps during an episode. We attacked all time steps just for a fair comparison. Second, we could not extensively investigate the performance of ACADIA under various continuous control environments of DRL.

ACKNOWLEDGEMENT

This work is partly supported by Virginia's Commonwealth Cyber Initiative (CCI) and NSF Grant 2107450.

REFERENCES

[1] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2017, pp. 39–57.

[2] T. Chen et al., "Gradient band-based adversarial training for generalized attack immunity of a3c path finding," *arXiv preprint arXiv:1807.06752*, 2018.

[3] F. Croce and M. Hein, "Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks," in *International Conference on Machine Learning*. PMLR, 2020, pp. 2206–2216.

[4] Y. Dong, et al.,, "Boosting adversarial attacks with momentum," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 9185–9193.

[5] A. M. et al., "Towards deep learning models resistant to adversarial attacks," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

[6] M. Fischer, M. Mirman, S. Stalder, and M. Vechev, "Online robustness training for deep reinforcement learning," *arXiv preprint arXiv:1911.00887*, 2019.

[7] I. J. e. Goodfellow, "Explaining and harnessing adversarial examples," *ICLR*, 2014.

[8] S. H. Huang, et al., "Adversarial attacks on neural network policies," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*.

[9] J. Kos and D. Song, "Delving into adversarial attacks on deep policies," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*.

[10] A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*.

[11] Y.-C. Lin, Z.-W. Hong, Y.-H. Liao, M.-L. Shih, M.-Y. Liu, and M. Sun, "Tactics of adversarial attack on deep reinforcement learning agents," in *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, ser. IJCAI'17. AAAI Press, 2017, p. 3756–3762.

[12] ——, "Tactics of adversarial attack on deep reinforcement learning agents," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, 2017, pp. 3756–3762.

[13] V. e. a. Mnih, "Playing Atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[14] V. Mnih, et al.,, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[15] T. Oikarinen, et al., "Robust deep reinforcement learning through adversarial loss," in *Advances in Neural Information Processing Systems*, 2021.

[16] X. Pan, C. Xiao, W. He, S. Yang, J. Peng, M. Sun, M. Liu, B. Li, and D. Song, "Characterizing attacks on deep reinforcement learning," in *21st International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2022, Auckland, New Zealand, May 9-13, 2022*.

[17] A. Pattanaik, et al., "Robust deep reinforcement learning with adversarial attacks," *arXiv preprint arXiv:1712.03632*, 2017.

[18] J. Sun, T. Zhang, X. Xie, L. Ma, Y. Zheng, K. Chen, and Y. Liu, "Stealthy and efficient adversarial attacks against deep reinforcement learning," *Proc. the AAAI Conference on Artificial Intelligence*, vol. 34, pp. 5883–5891.

[19] F. Tramèr, et al.,, "Ensemble adversarial training: Attacks and defenses," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

[20] E. Tretschk, S. J. Oh, and M. Fritz, "Sequential attacks on agents for long-term adversarial goals," *arXiv preprint arXiv:1805.12487*, 2018.

[21] Y. Xiao, C.-M. Pun, and B. Liu, "Adversarial example generation with adaptive gradient search for single and ensemble deep neural network," *Information Sciences*, vol. 528, pp. 147–167, 2020.

[22] C. Xie, Z. Zhang, Y. Zhou, S. Bai, J. Wang, Z. Ren, and A. L. Yuille, "Improving transferability of adversarial examples with input diversity," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2730–2739.

[23] H. Yin, H. Zhang, J. Wang, and R. Dou, "Improving the transferability of adversarial examples with the adam optimizer," *arXiv preprint arXiv:2012.00567*, 2020.

[24] X. Yuan et al.,, "Adversarial examples: Attacks and defenses for deep learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 9, pp. 2805–2824, 2019.

[25] H. Zhang, H. Chen, D. S. Boning, and C.-J. Hsieh, "Robust reinforcement learning on state observations with learned optimal adversary," in *International Conference on Learning Representations*, 2021.

[26] H. Zhang, et al.,, "Robust deep reinforcement learning against adversarial perturbations on state observations," in *Advances in Neural Information Processing Systems*, 2020.